

**REMARKS/ARGUMENTS**

This Amendment and the following remarks are intended to fully respond to the Final Office Action mailed February 1, 2006. In that Office Action, claims 1-51 were examined, and all claims were rejected. More specifically, claims 1-20, 25-30, 33-38, 40-43, and 47-50 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Blais et al. (USPN 6,505,344); and claims 21-24, 31, 32, 39, 44-46, and 51 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Blais in view of Pinter et al. (USPN 6,457,203). Reconsideration of these rejections, as they might apply to the original and amended claims in view of these remarks, is respectfully requested.

In this Response, claims 1, 25, 34, 35, 40 and 47 have been amended to clarify that the proven thread-specific data is allocated to a thread-specific heap. No claims have been added or canceled.

**Request For Withdrawal of Final Office Action**

The Applicants respectfully request the withdrawal of the Final Office Action. The Applicants first point out that in the Office Action Summary, the Examiner has indicated an objection to the specification. However, there is no mention in the Detailed Action of the specific portion(s) to which the Examiner objects. Therefore, the Applicants respectfully request clarification as to the specific portions of the specification to which the Examiner objects, and an opportunity to make any necessary amendments to the specification by withdrawing the Final Office Action.

Moreover, the Applicants point out that the Examiner has cited new references in the Final Office Action, namely the Choi et al. references (Escape Analysis for Java and U.S. Patent No. 6,381,738). Although, the Applicants realize that the Examiner cites the new references in maintaining the same ground of rejection from the previous Office Action, the Applicants have not previously had an opportunity to address these references. Applicants kindly point out that an “examiner should never lose sight of the fact that in every case the applicant is entitled to a full and fair hearing. . . .” *Manual of Patent Examining Procedure* (MPEP), § 706.07. For this additional reason, the Applicants kindly request that the Final Office Action be withdrawn.

**Claim Rejections**

The Examiner maintains the previous rejection of claims 1-20, 25-30, 33-38, 40-43 and 47-50 under 35 U.S.C. § 102(e) as being anticipated by Blais. The Examiner cites two new references in support of the rejection namely U.S. Patent No. 6,381,738 to Choi et al. (hereinafter the “Choi patent”), and Escape Analysis for Java to Choi et al. (hereinafter the “Choi reference”).

*Inter alia*, newly amended independent claims 1, 25, 34, 35, 40 and 47 recite analyzing a program to distinguish proven thread-specific data from shared data, and allocating the proven thread specific data to a thread-specific heap. None of the references cited by the Examiner disclose these elements of the claims.

The Examiner asserts that the newly cited Choi patent establishes that a region-based approach to allocating objects to a heap is regarded as equivalent to performing stack allocation of data. However, the Applicants kindly submit that the Examiner gives the Choi patent a broad reading that is not supported by the disclosures in the Choi patent. The specific language in the Choi patent cited by the Examiner states that “[a]n alternate approach to stack allocation is to use a region in the heap *for allocating objects whose lifetimes are bounded by the lifetime of the stack frame*, and deallocating the entire region when the corresponding procedure returns.” *Choi Patent*, Col. 1, line 65-col. 2, line 5 (emphasis added). Thus, the Examiner’s asserted equivalency only applies to those objects that are limited by the lifetime of the stack frame, in other words to a single procedure or method. This is necessary because the specified region in the heap is deallocated when the procedure returns, just like the situation in an invocation stack frame.

However, for those objects that may be referenced beyond a single stack frame, such as objects that may be reached by multiple stack frames executed in a thread, this equivalency does not apply. Indeed, the equivalency cannot apply because if the entire region were deallocated after a procedure returns, the object would not be available for other procedures that reference it. Accordingly, allocating thread-specific data (as opposed to data bounded by a method) to thread-specific heaps is not equivalent to allocating objects to an invocation stack frame, and as detailed in the previous office action response is a distinction between the independent claims and the Blais reference. For this reason alone, the independent claims are allowable over the cited references.

Moreover, even assuming *arguendo* that the equivalency asserted by the Examiner is correct, which as stated above the Applicants dispute, none of the references and in particular Blais would still anticipate the independent claims. The claims require analyzing a target program during code compilation to distinguish between proven thread-specific data and shared data. The proven thread-specific data is then allocated to a thread-specific heap. To anticipate the claims using the Examiner's asserted equivalency, there would have to be some disclosure in the references that teach that all thread specific data is allocated to the invocation stack frame. As described below, neither Choi nor Blais teaches or suggests such a method, and indeed both teach away from such a method.

Firstly, Applicants point out that Choi describes the objects marked as ArgEscape as being thread specific objects. Choi states that "ArgEscape [objects] are local to the thread in which they are created . . . ." *Choi Reference*, page 3. The allocation of the ArgEscape objects by Choi and Blais highlight the differences among Choi, Blais and the independent claims.

Choi merely allocates the ArgEscape objects to a shared heap with the Global Escape objects. As stated by Blais, "[Choi's] prior art escape analysis treats the global escape and arg escape cases the same, forcing them to be allocated from the heap." *Blais*, col. 8, lines 41-43. Thus, Choi allocates thread-specific objects to a shared heap, which is not that same as the invocation stack frame or a thread-specific heap.

Blais performs further analysis on the ArgEscape objects to allocate "some of these arg escape objects to a method's invocation stack frame instead of forcing all arg escape objects to be allocated from the heap." *Blais*, col. 8, lines 47-49. Thus, Blais allocates some thread-specific objects to the invocation stack frame and others to a shared heap. Based on the Examiner's asserted equivalency, Blais would have to teach allocating all the ArgEscape objects to the invocation stack frame, however this is not the case. Rather, Blais' entire invention is directed at a "further analysis . . . to determine whether the [arg escape] object can be allocated on an invoking method's stack or must be allocated from the heap." *Blais*, col. 3, lines 3-6.

In contrast to both Choi and Blais, the independent claims allocate the proven thread-specific data to a thread-specific heap. The thread specific data is not allocated to a shared heap like Choi. Nor is some of the proven thread-specific data allocated to the invocation stack frame and other proven thread-specific data to a shared heap, like Blais. For these additional reasons, independent claims 1, 25, 34, 35, 40 and 47 are allowable over the cited Blais and Choi

references. Claims 2-24, 26-33, 36-39 and 41-51 are also allowable, as they depend directly or indirectly from the independent claims.

Claims 21-24, 31, 32, 39, 44-46, and 51 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Blais in view of Pinter et al. (USPN 6,457,203).

As described above, Blais and Choi do not disclose allocating proven thread-specific objects to a thread specific heap, and Pinter et al. does not compensate for this deficiency.

Claims 21-24, 31, 32, 39, 44-46, and 51 depend, directly or indirectly, from one of claims 1, 25, 34, 35, 40 and 47, and are therefore allowable for the same reasons described above.

**Conclusion**

This Amendment fully responds to the Final Office Action mailed on February 1, 2006. Still, that Office Action may contain arguments and rejections and that are not directly addressed by this Amendment due to the fact that they are rendered moot in light of the preceding arguments in favor of patentability. Hence, failure of this Amendment to directly address an argument raised in the Office Action should not be taken as an indication that the Applicant believes the argument has merit. Furthermore, the claims of the present application may include other elements, not discussed in this Amendment, which are not shown, taught, or otherwise suggested by the art of record. Accordingly, the preceding arguments in favor of patentability are advanced without prejudice to other bases of patentability.

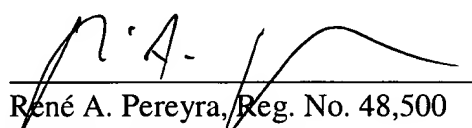
It is believed that no further fees are due with this Response. However, the Commissioner is hereby authorized to charge any deficiencies or credit any overpayment with respect to this patent application to deposit account number 13-2725.

In light of the above remarks and amendments, it is believed that the application is now in condition for allowance and such action is respectfully requested. Should any additional issues need to be resolved, the Examiner is requested to telephone the undersigned to attempt to resolve those issues.

Respectfully submitted,

march 31, 2006  
Date



  
René A. Pereyra, Reg. No. 48,500  
Merchant & Gould P.C.  
P.O. Box 2903  
Minneapolis, MN 55402-0903  
(303) 357-1651  
(303) 357-1671 (fax)